Module 4: Springboot and Spring Framework – II

Table of Contents

4. Overview 1

4.1 Endpoints 1

4.1.1 Fetch Fees by Student ID.. 1

4.1.2 Pay Fees.. 2

4.2 Implementation Details.. 3

4.3 Initialization of Sample Data 3

4. Overview

The Fees Management Microservice (**feesms**) is a component of the School Management Software designed to handle fee-related operations for students.

GIT URL: https://github.com/vigneshpshetty/edureka-microservice-course/tree/main/m4-feesms

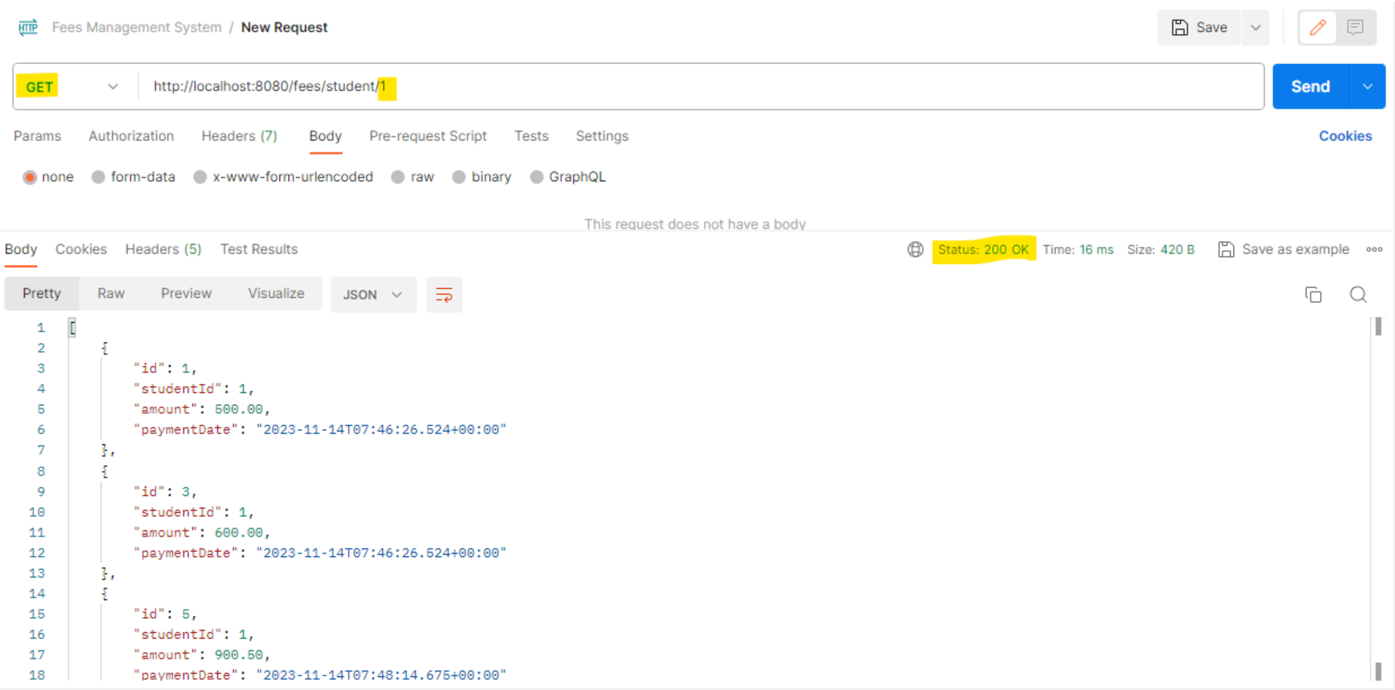
4.1 Endpoints

4.1.1 Fetch Fees by Student ID

- **Endpoint: GET /fees/student/{studentId}**
- **Description:** Retrieve all fees paid by a specific student.
- **Request Example: /fees/student/1**
- **Response Example:**

```
[
    {
        "id": 1,
        "studentId": 1,
        "amount": 500.00,
        "paymentDate": "2023-11-14T07:46:26.524+00:00"
    },
    {
        "id": 3,
        "studentId": 1,
        "amount": 600.00,
        "paymentDate": "2023-11-14T07:46:26.524+00:00"
    },
    {
        "id": 5,
        "studentId": 1,
        "amount": 900.50,
        "paymentDate": "2023-11-14T07:48:14.675+00:00"
    }
]
```

Screenshots



GET http://localhost:8080/fees/student/1

This request does not have a body

Status: 200 OK Time: 16 ms Size: 420 B

```
[
    {
        "id": 1,
        "studentId": 1,
        "amount": 500.00,
        "paymentDate": "2023-11-14T07:46:26.524+00:00"
    },
    {
        "id": 3,
        "studentId": 1,
        "amount": 600.00,
        "paymentDate": "2023-11-14T07:46:26.524+00:00"
    },
    {
        "id": 5,
        "studentId": 1,
        "amount": 900.50,
        "paymentDate": "2023-11-14T07:48:14.675+00:00"
```

4.1.2 Pay Fees

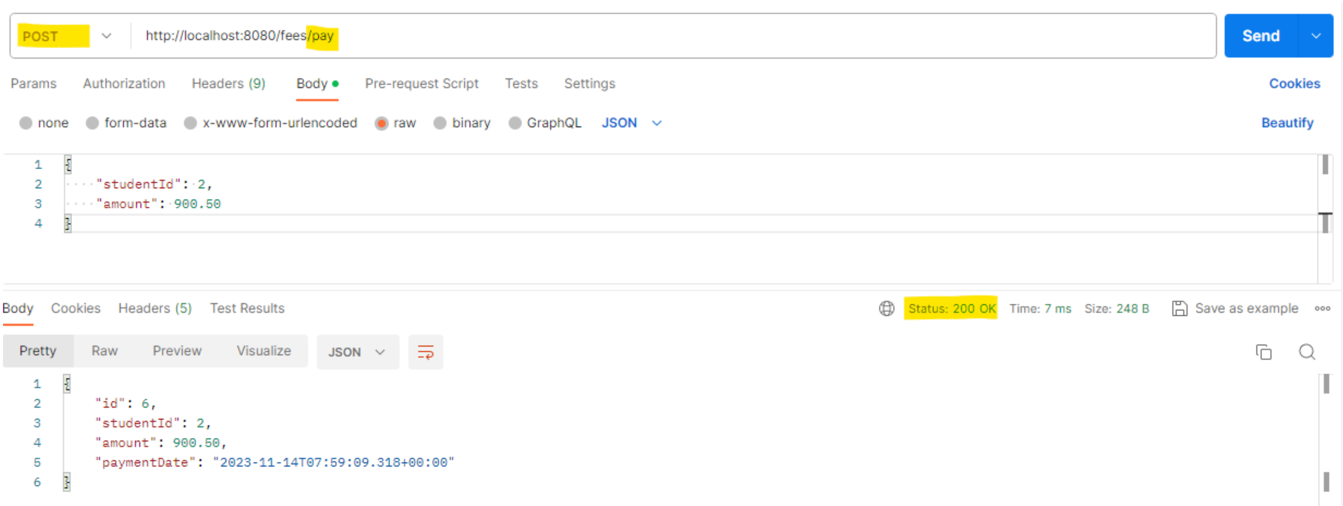
- **Endpoint: POST /fees/pay**
- **Description:** Submit a payment for a student's fees.
- **Request Example:**

```
{
    "studentId": 2,
    "amount": 1000.50
}
```

- **Response Example:**

```
{
    "id": 7,
    "studentId": 2,
    "amount": 1000.50,
    "paymentDate": "2023-11-14T08:00:56.637+00:00"
}
```

Screenshots



POST http://localhost:8080/fees/pay

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
    "studentId": 2,
    "amount": 900.50
}
```

Status: 200 OK

JSON

```
{
    "id": 6,
    "studentId": 2,
    "amount": 900.50,
    "paymentDate": "2023-11-14T07:59:09.318+00:00"
}
```

4.2 Implementation Details

- **Controller Class: FeesController.java**
    - Manages HTTP endpoints related to fee operations.
- **Service Class: FeesService.java**
    - Implements business logic for fee-related operations.
- **DTO Class: FeePaymentRequest.java**
    - Represents the data structure for fee payment requests.
- **Entity Class: Fee.java**
    - Represents the data structure for fee records.

4.3 Initialization of Sample Data

- The **@PostConstruct** method in **FeesService** initializes sample fee records when the application starts.

```
@PostConstruct
private void initializeData() {
    // Insert sample data when the application starts
    Fee fee1 = new Fee(1L, BigDecimal.valueOf(500), new Date());
    Fee fee2 = new Fee(2L, BigDecimal.valueOf(750), new Date());
    Fee fee3 = new Fee(1L, BigDecimal.valueOf(600), new Date());

    feeRepository.saveAll(List.of(fee1, fee2, fee3));
}
```